IN THE CLAIMS

1. (Currently Amended) A method for processing events in a managed information system comprising:

defining a plurality of events, the plurality of events associated with a genericizing reference, the genericizing reference inclusive of the plurality of events and each of the events associated with an event specific class having event data indicative of event specific parameters, each of the events defined independently of an underlying delivery infrastructure;

receiving an event subscription containing an event identity for an event, the event corresponding to reportable occurrences in the managed information system;

associating the event identity with an event handler responsive to the event by creating a mapping of the event identity to the event handler;

receiving a publication of the event; ~~and~~

traversing, in response to the publication, the mapping of the event identity to an indication of the corresponding associated handler, the traversing operable to enable the module including the event handler if the module is disabled at the time of publishing the event, such that

the enabling of modules corresponds to activation of a corresponding component by an activation mechanism, and disabling corresponding to deactivation of the corresponding component by the activation mechanism,

the activation and deactivation operations being operable to reduce memory consumption by inactive components and provide selective invocation to maintain availability of the component, the enabling and disabling performed at a level of granularity of the modules, each of the modules corresponding to a component and operable be enabled and disabled by activation and deactivation of the corresponding component, the event data including event variables generated and passed by the

publisher of the event and subscriber instantiated variables generated by the state information of the subscriber; and

selectively enabling, if the module including the corresponding event handler is disabled, the module for enabling the event handler for receipt and subsequent processing of the published event.

2. (Original) The method of claim 1 further comprising:

identifying, using the associated event identity, the particular handler corresponding to the subscribed event in the enabled module including the handler;

invoking, using the state of the enabled module, the event handler.

3. (Original) The method of claim 1 wherein traversing further comprises:

receiving the event publication according to a genericizing reference; and

identifying an event specific class corresponding to the event, the event specific class transparent to the mapping via the genericizing reference and operative to distinguish the received events from other events.

4. (Currently Amended) The method of claim 1 wherein traversing further comprises:

determining if a particular module including the corresponding event handler is enabled~~; and~~

~~selectively enabling, if the module including the corresponding event handler is disabled, the module for enabling the event handler for receipt and subsequent processing of the published event~~.

5. (Original) The method of claim 4 further comprising, following selective enabling of the module containing the corresponding handler:

determining the mapping of the enabled module and corresponding event handler; and

invoking the module including the corresponding event handler via the mapping.

6. (Original) The method of claim 1 wherein traversing further comprises

identifying the event in a persistent event mapping, the persistent event mapping indicative of modules containing event handlers associated with the event; and

dispatching, in the identified modules, the associated event handlers.

7. (Original) The method of claim 6 wherein dispatching further comprises:

selectively receiving an enablement indication in response to traversing in the persistent event mapping;

identifying, in a local event mapping for the enabled module, subscriber entities including handlers associated with the mapped event; and

invoking, in the mapped module, the identified subscribers including associated handlers.

8. (Original) The method of claim 6 wherein the persistent mapping of the event is operable to maintain the event independently of individual modules referencing the event, the independent maintenance operable to avoid copy constructors of the event for enabling successive references to the same event.

9. (Original) The method of claim 2 further comprising:

disabling a publishing component performing the publishing prior to invoking the module including the event handler; and

completing the invocation of the corresponding handler while the publishing component remains disabled.

Claims 10-11. (Canceled)

12. (Previously Presented) The method of claim 1 further comprising enumerating a plurality of events, wherein the event further comprises the plurality of events associated with the common genericizing reference inclusive of the plurality of events, and wherein receiving the event subscription avoids event specific code generation of code and code fragments associated with the specific event.

13. (Previously Presented) The method of claim 1 wherein the common genericizing reference and associated event specific class avoids event specific stubs and references related to the event specific class.

14. (Original) The method of claim 1 wherein a subscribing software entity issuing the received event subscription becomes disabled following the subscription until an occurrence and subsequent publication of the event.

15. (Original) The method of claim 1 wherein publication of the event is operable to enable a plurality of subscribing software entities, each subscribing entity including a particular responsive event handler for handling that event.

16. (Original) The method of claim 6 wherein traversing further comprises indexing, in the persistent mapping via the event identity, a persistent reference to the modules including the event handlers associated with the event, the persistent reference operable to identify a handler independently of enablement of the module containing the associated event handler.

17. (Original) The method of claim 1 wherein associating the event identity by creating a mapping with the event handler further comprises:

creating, via a component event service, a local mapping entry in the component event map having a reference to the subscriber entity including the corresponding event handler, and

creating a persistent mapping entry corresponding to the component including the corresponding event handler, the persistent mapping entry operable to trigger selective enablement of the handling component by a plurality of subscribing entities, wherein mapping further comprises:

identifying at least one of the persistent mapping entries corresponding to the published event, each of the mapping entries indicative of a module; and

identifying, via the local event map in the indicated modules, a plurality of subscribers including the corresponding event handlers in the identified modules associated with the event.

Claims 18-19. (Canceled)

20. (Previously Presented) The method of claim 1 wherein activation and deactivation further comprises identifying, in a component server in communication with the shared memory portion, when to activate and deactivate components based on information in the persistent event map in the shared memory portion, and further for determining when to store the information in the component server rather than shared memory if no other component servers reference the information.

21. (Previously Presented) The method of claim 1 wherein each of the modules is operable to include a plurality of threads, and disabling is performed by a thread manager operable to gracefully terminate each of the threads prior to deactivation, deactivation occurring by informing each of the threads of the termination and computing when each thread has attained a termination point.

22. (Original) The method of claim 1 wherein associating the event identity with an event handler occurs in a native language of the event handler

and corresponding subscriber, and avoids a corresponding definition in an external interface language, the external interface language for generating event specific code.

23. (Previously Presented) The method of claim 22 wherein the external interface language is an Interface Definition Language .

24. (Original) The method of claim 23 wherein associating the event identity with an event handler further comprises generating a local mapping via a component event service identifying a subscribing entity including an event handler corresponding to the event identity, and

generating a persistent event mapping identifying the module including the event handler corresponding to the event identity.

25. (Original) The method of claim 24 further comprising selectively generating the persistent event mapping via a strategized allocator if the associating of the event identity occurs in a single module.

26. (Original) The method of claim 1 further comprising a smart event pointer class, wherein publication further comprises:

identifying a plurality of modules having at least one of subscribers and publishers including an event;

referencing, via a reference counting semaphore in the smart event point class, each of the instantiations of the event in a particular module;

identifying, via the reference counting semaphore, when instantiations of the event have completed instructions corresponding to the event handler in each of the particular modules; and

deallocating the event from the persistent event mapping when each module including instantiations of the event has identified completion of instructions corresponding to the event handler.

27. (Original) The method of claim 26 further comprising:

maintaining references to instantiations of the event across multiple modules via the reference counting semaphore, the multiple modules corresponding to a plurality of dynamic allocation mechanisms; and

deallocating, for all of the referenced dynamic allocation mechanisms, the instantiations of the event when each of the multiple modules completes processing of the event.

28. (Original) The method of claim 1 wherein the subscribing entity, publishing entity, and handling entity are user software entities responsive to the local event service for execution, activation, and inactivation.

29. (Canceled)

30. (Previously Presented) The method of claim 2 wherein invoking further comprises selective activation of components including associated event handlers, the event handlers in inactive components responding to the dispatch upon activation.

31. (Original) The method of claim 30 wherein the components including the event handlers need not be active during the event publication, the inactive event handlers being invoked accordingly in response to the dispatching, wherein unavailable event handlers consume fewer resources than available event handlers.

32. (Canceled)

33. (Currently Amended) A method for interprocess communication comprising:

defining an event indicative of a reportable occurrence in the information retrieval environment, the event corresponding to a common class associated

with a plurality of events, the plurality of events associated with a genericizing reference, the genericizing reference inclusive of the plurality of events and each of the events associated with an event specific class having event data indicative of event specific parameters, each of the events defined independently of an underlying delivery infrastructure;

subscribing, from a local subscriber, to the event for requesting reporting of the event, subscribing further including specifying an event handler operable to process the event, the local subscriber issuing the received event subscription becoming disabled following the subscription until an occurrence and subsequent publication of the event;

registering an indication of the subscription in a local map providing a local mapping operative for associating events and event handlers in a local process for invoking the event handler responsively to an occurrence of the defined event;

registering an indication of the subscription in a global map providing a persistent mapping operative for associating events with event handlers, and further operable to invoke event handlers in remote servers;

receiving a publication of the event from a monitoring component;

dispatching an indication of the publication to local subscribers;

propagating an indication of the publication to the global event service; and

invoking a module including the specified event handlers in response to the dispatching and propagating the indication of event publication, invoking including indexing, in the persistent mapping via the event identity, a persistent reference to the modules including the event handlers associated with the event, the persistent reference operable to identify a handler independently of enablement of the module containing the associated event handler,

enabling modules corresponding to activation of a corresponding component by an activation mechanism; ~~and~~

disabling corresponding to deactivation of the corresponding component by the activation mechanism, the activation and deactivation operations operable

to reduce memory consumption by inactive components and provide selective invocation to maintain availability of the component, the event data including event variables generated and passed by the publisher of the event and subscriber instantiated variables generated by the state information of the subscriber; and

selectively enabling, if the module including the corresponding event handler is disabled, the module for enabling the event handler for receipt and subsequent processing of the published event.

34. (Currently Amended) An information processing device for processing events in a managed information system comprising:

an interface for defining a plurality of events, the plurality of events associated with a genericizing reference, the genericizing reference inclusive of the plurality of events and each of the events associated with an event specific class having event data indicative of event specific parameters, each of the events defined independently of an underlying delivery infrastructure;

a processor operable to perform a set of instructions according to a predetermined sequence;

a memory operable to store the instructions and corresponding data items;

an interface operable to transmit and receive, based on the instructions, commands and data to other information processing devices in the managed information system;

an event service operable to receive an event subscription containing an event identity for an event, the event corresponding to reportable occurrences in the managed information system;

an event map operable to associate the event identity with an event handler responsive to the event by creating a mapping of the event identity to the event handler; ~~and~~

a subscriber having an event handler operable to receive a publication of the event, the event service further operable to traverse, in response to the publication of the event, the mapping of the event identity to an indication of the

corresponding associated event handler, the traversing operable to enable the module including the handler if the module is disabled at the time of publishing the event, traversing further including:

identifying the event in a persistent event mapping, the persistent event mapping indicative of modules containing event handlers associated with the event; and

dispatching, in the identified modules, the associated event handlers, dispatching comprising:

selectively receiving an enablement indication in response to traversing in the persistent event mapping;

identifying, in a local event mapping for the enabled module, subscriber entities including handlers associated with the mapped event; and

invoking, in the mapped module, the identified subscribers including associated handlers,

the event data including event variables generated and passed by the publisher of the event and subscriber instantiated variables generated by the state information of the subscriber; and

an activation manager operable to determine if a particular module including the corresponding event handler is enabled, and further operable to selectively enable, if the module including the corresponding event handler is disabled, the module for enabling the event handler for receipt and subsequent processing of the published event.

35. (Original) The method of claim 34 wherein the event service further includes:

a component event service operable to identify, using the associated event identity, the particular handler corresponding to the subscribed event in the enabled module including the handler; and

a local event map in the component event service operable for invoking, using the state of the enabled module, the event handler.

36. (Original) The method of claim 34 wherein the event service further includes a global event service, the global event service operable to:

receive the event publication according to a genericizing reference; and

identify an event specific class corresponding to the event, the event specific class transparent to the mapping via the genericizing reference and operative to distinguish the received events from other events.

37. (Canceled).

38. (Currently Amended) The method of claim 3437 wherein the event service is further operable to, following selective enabling of the module containing the corresponding handler:

determine the mapping of the enabled module and corresponding event handler; and

invoke the module including the corresponding event handler via the mapping.

39. (Original) The method of claim 34 wherein the event service further comprises:

a persistent event map operable to identify the published event, the persistent event mapping indicative of modules containing event handlers associated with the event; and

a local event map operable to dispatch, in the identified modules, the associated event handlers.

40. (Original) The method of claim 39 further comprises:

a module containing the event handler, the module operable to selectively receive an enablement indication in response to traversing in the persistent event mapping, the module further including the local event map indicative of subscriber entities including handlers associated with the mapped

event, the module operable to invoke, via the local event map, the identified subscribers including associated handlers.

41. (Original) The method of claim 39 wherein the persistent map of the event is operable to maintain the event independently of individual modules referencing the event, the independent maintenance operable to avoid copy constructors of the event for enabling successive references to the same event.

42. (Original) The method of claim 34 further comprising a plurality of events, the plurality of events associated with a genericizing reference, the genericizing reference inclusive of the plurality of events and each of the events associated with an event specific class having event data indicative of event specific parameters.

43. (Original) The method of claim 42 wherein the event data includes event variables generated and passed by the publisher of the event and subscriber instantiated variables generated by the state information of the subscriber.

44. (Original) The method of claim 39 wherein traversing further comprises indexing, in the persistent mapping via the event identity, a persistent reference to the modules including the event handlers associated with the event, the persistent reference operable to identify a handler independently of enablement of the module containing the associated event handler.

45. (Original) The method of claim 34 wherein the event service further includes:

a component event service operable to generate the local mapping entry in the component event map having a reference to the subscriber entity including the corresponding event handler, and

a persistent event map operable to store a persistent mapping entry corresponding to the module including the corresponding event handler, the persistent mapping entry operable to trigger selective enablement of the handling component by a plurality of subscribing entities, the persistent event map further operable to identify at least one of the persistent mapping entries corresponding to the published event, each of the mapping entries indicative of a module;

a local event map operable to a plurality of subscribers including the corresponding event handlers in the identified modules associated with the event.

46. (Original) The method of claim 37 further comprising a thread manager, wherein each of the modules is operable to include a plurality of threads, and wherein the thread manager is operable to perform graceful termination each of the threads prior to deactivation, deactivation occurring by informing each of the threads of the termination and computing when each thread has attained a termination point.

47. (Original) The method of claim 34 wherein the association between the event identity and the corresponding event handler further comprises:

a local mapping in a local event map identifying a subscribing entity including an event handler corresponding to the event identity, and

a persistent event mapping in a shared memory portion identifying the module including the event handler corresponding to the event identity.

48. (Original) The method of claim 47 further comprising a strategizing allocator operable to selectively generate the persistent event mapping if the associating of the event identity occurs in a single module.

49. (Currently Amended) A computer program product having a computer readable storage medium operable to store computer program logic embodied in computer program code as a set of computer instructions encoded thereon for performing a method of processing events in a managed information system, the method comprising:

~~computer program code for~~ defining a plurality of events, the plurality of events associated with a genericizing reference, the genericizing reference inclusive of the plurality of events and each of the events associated with an event specific class having event data indicative of event specific parameters, each of the events defined independently of an underlying delivery infrastructure;

~~computer program code for~~ receiving an event subscription containing an event identity for an event, the event corresponding to reportable occurrences in the managed information system;

~~computer program code for~~ associating the event identity with an event handler responsive to the event by creating a mapping of the event identity to the event handler;

~~computer program code for~~ receiving a publication of the event; and

~~computer program code for~~ traversing, in response to the publication, the mapping of the event identity to an indication of the corresponding associated handler, the traversing operable to enable the module including the event handler if the module is disabled at the time of publishing the event, enabling modules corresponding to activation of a corresponding component by an activation mechanism, and disabling corresponding to deactivation of the corresponding component by the activation mechanism, the activation and deactivation operations operable to reduce memory consumption by inactive components and provide selective invocation to maintain availability of the component, the enabling and disabling performed at a level of granularity of the modules, each of the modules corresponding to a component and operable be enabled and disabled by activation and deactivation of the corresponding component,

the event data including event variables generated and passed by the publisher of the event and subscriber instantiated variables generated by the state information of the subscriber; and

selectively enabling, if the module including the corresponding event handler is disabled, the module for enabling the event handler for receipt and subsequent processing of the published event.

50. (Canceled)

51. (Currently Amended) An information processing device for processing events in a managed information system comprising:

means for defining a plurality of events, the plurality of events associated with a genericizing reference, the genericizing reference inclusive of the plurality of events and each of the events associated with an event specific class having event data indicative of event specific parameters, each of the events defined independently of an underlying delivery infrastructure;

means for receiving an event subscription containing an event identity for an event, the event corresponding to reportable occurrences in the managed information system;

means for associating the event identity with an event handler responsive to the event by creating a mapping of the event identity to the event handler, the association between the event identity and the corresponding event handler further comprising:

a local mapping in a local event map identifying a subscribing entity including an event handler corresponding to the event identity, and

a persistent event mapping in a shared memory portion identifying the module including the event handler corresponding to the event identity;

means for receiving a publication of the event; and

means for traversing, in response to the publication, the mapping of the event identity to an indication of the corresponding associated handler, the traversing operable to enable the module including the event handler if the module is disabled at the time of publishing the event, the means for traversing further including:

means for identifying the event in a persistent event mapping, the persistent event mapping indicative of modules containing event handlers associated with the event; and

means for dispatching, in the identified modules, the associated event handlers, dispatching comprising:

means for selectively receiving an enablement indication in response to traversing in the persistent event mapping;

means for identifying, in a local event mapping for the enabled module, subscriber entities including handlers associated with the mapped event; and

means for invoking, in the mapped module, the identified subscribers including associated handlers,

the event data including event variables generated and passed by the publisher of the event and subscriber instantiated variables generated by the state information of the subscriber; and

means for selectively enabling, if the module including the corresponding event handler is disabled, the module for enabling the event handler for receipt and subsequent processing of the published event.

52. (New) The method of claim 1 wherein activating further comprises:

loading a module, the module containing the instructions defining the component, from a nonexecutable memory area to an executable memory area for execution of the instructions; and

deactivating further comprises unlinking the loaded module by disassociating the loaded module from the executable memory area.

53. (New) The method of claim 52 wherein the loaded modules are sequences of instructions defining at least one thread, all of the threads collectively activated and deactivated via the component defined by the module.

54. (New) The method of claim 53 wherein selective activation further comprises:

detecting, from a loaded component defined by a detecting software entity, occurrence of an instance corresponding to an event handler; and

invoking the event handler by activating the component including the event handler.